View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Elsevier - Publisher Connector

Available online at www.sciencedirect.com

SciVerse ScienceDirect

Physics Procedia 33 (2012) 1640 – 1646

Physics Procedia

2012 International Conference on Medical Physics and Biomedical Engineering

Design and Implementation of KSP on the Next Generation Cryptography API*

Zhang Lina

School of Computer Science and Technology
Xi'an University of Science and Technology
Xi'an Shanxi China
zhangln@xust.edu.cn

Abstract

With good seamless connectivity and higher safety, KSP (Key Storage Providers) is the inexorable trend of security requirements and development to take the place of CSP (Cryptographic Service Provider). But the study on KSP has just started in our country, and almost no reports of its implementation can be found. Based on the analysis of function modules and the architecture of Cryptography API (Next Generation (CNG)), this paper discusses the design and implementation of KSP（key storage providers） based on smart card in detail, and an example is also presented to illustrate how to use KSP in Windows Vista.

© 2011 Published by Elsevier Ltd. Selection and/or peer-review under responsibility of [name organizer]
Open access under CC BY-NC-ND license.

Keywords: Smart Card, CNG, CSP, KSP

1. Introduction

With the development and popularization of Internet, the security problems of the Internet are becoming increasingly serious. It is necessary to establish some security policy and mechanism to satisfy the information security requirements of confidentiality, integrality and non-repudiation which are the kernel problems in cryptography. Judging from current situation, the work for network information security is far from enough. Because the cryptographic algorithm needs more mathematical foundation and general developers don't have cryptography background, they have to choose some simple encryption technology to implement security service and mechanism with lower security.

Based on this situation, Microsoft Windows' OS provides users with all application program interfaces for cryptography by CSP (Cryptographic Service Provider) [1]. Application developers can use the functions in CryptoAPI which works with a number of cryptographic service providers without

* This work is partially supported by The Cultivate Fund of Xi'an University of Science and Technology(No. 2010031)

1875-3892 © 2012 Published by Elsevier B.V. Selection and/or peer review under responsibility of ICMPBE International Committee.
Open access under CC BY-NC-ND license. doi:10.1016/j.phpro.2012.05.264

knowing details of the underlying implementation, in much the same way as they can use a graphics library without knowing anything about the particular graphics hardware configuration. As smart card has becoming the mainstream of data storage materials, the encrypt server based on CSP could provide higher security by combined with smart card. But with the fast development of information technology and improvement of the computer application technology, CSP could hardly satisfy the application demand for its security and agility. CSP could only afford RSA, DSA, DH and simple symmetric algorithm such as DES and RC4 without user-defined algorithms.

Now Cryptography API: Next Generation (CNG) [2], put forward by Microsoft, is the long-term replacement for the CryptoAPI. CNG is designed to be extensible at many levels and support for all required algorithms including elliptic curve cryptography (ECC) and any other user-defined algorithm[3][4][5]. The biggest advantage and features of CNG is that it provides a model for private key storage that allows adaptation to the demands of creating applications with cryptography features such as public or private key encryption, as well as the demands of the storage of key material. An application could access the key storage providers (KSPs) on the system through the key storage router which is implemented in Ncrypt.dll to complete their functional requirement, and KSP could conceal details, such as key isolation, from both the application and the storage provider itself.

Currently the study on KSP has just started and there are almost no research conclusions or implementation details based on KSP framework. The design and implementation of KSP based on smart card is discussed in this paper, and then a typical application example is also presented.

2. BRIEF introduction of CNG

From the view of function, CNG could be divided into six algorithm types, including random number generator, Hash, symmetric encryption, asymmetric encryption, signature and secret agreement. Each algorithm or class of algorithms exposes its own primitive API. Multiple implementations of a given algorithm can be installed at the same time; however, only one implementation will be the default one at any given time. The ECC signature is chosen to complete in KSP of this paper.

Then from the view of structure, CNG could be divided into two quite different basic classifications, including Cryptographic Algorithm Provider (CAP) and KSP. Generally CAP implements some cryptography operations unrelated with private key, such as hash algorithm and symmetric encryption algorithm whose key was generated for temporary session. KSP is mainly used to store private key, as well as perform some cryptographic operations related with private key such as signatures, and it could guarantee the security of private keys because once the private key was stored, it will not be read. In the extensive use of PKI system[6], it needs to use digital certificates, encryption/decryption algorithms and digital signature technology to satisfy the information security requirements of confidentiality, integrality and incontestability，so the way to store the public key and private key with certificate and private key information is very important. Now the general way is to store the private key and certificate[7] in a smart card[8] to enhance its confidentiality and identity, meanwhile KSP in CNG could just provides corresponding model and interface to connect and communicate with smart card，then the private key could be safely stored in the smart card with specific function such as encrypt or signature to keep the safety, reliability and specificity of application. The structure of CNG is shown in Figure 1:

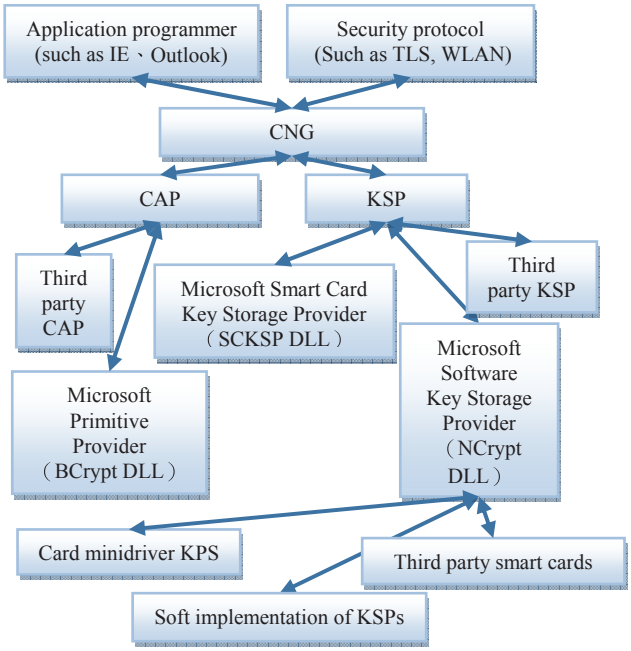

Fig. 1 The structure of CNG

3. THE design and Implementation of KSP

3.1The internal structure of KSP

There are two important structures of MAP in the design, one is ProvMap<phandle,provider> and the other is KeyMap<khandle,key>.The relationship between the two MAPs and their underlying structures are shown in Fig.2.

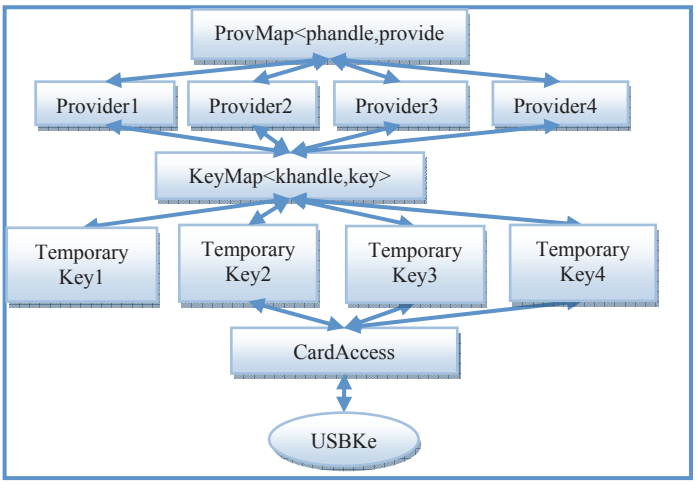

Fig. 2 the structure of MAP

The design of the KSP structure can be divided into the following four classes: Ksp_Main which is the outermost layer and communicates with the interface; Ksp_provider which completes the parsing and calling of internal command; KSP_Key which completes the related functions with key; KSP_CardAccess which interacts with a third party of DLL. The chief specification is showed as follows:

1) The class of KSP_Main

KSP_Main class is the interface class of system and should be identical with the interface described in the document of MS KSP. The command parsing and various types of function calls have been accomplished in the inner of the interface functions which could be easy to be directly called or registered to the system by corresponding dll.

2) The class of KSP_Provider

Application is used to call OpenProvider which would generate an object of KSP_Provider class and the object would be added to the ProvMap. ProvMap is static and superb simultaneously, which consists of the handle, and a pair of objects of provider will get the object by the handle. The major functions of the object are to parse command, set properties, generate key, open key and delete key as provider of operating service, and many providers could be opened as well at the same time, which is frequently needed in applied cases. Mere persistent key could be written into USB key, whereas the temporary key, as a verifying key, could not be written into USB key. Each opened or generated key would be joined in the superb one and static KeyMap, and thus KeyHandle could find the corresponding object of key.

3) The class of KSP_Key

The object of the key within KSP_key class obtains some functions of operating keys such as setting the properties of key, importing and exporting key, signing and verifying etc. Every persistent key reflects one logical storage unit in key container of USB key, including physical storage files such as state, name of key, private key, public key, certification and property, etc.

4) The class of KSP_CardAccess

KSP_Card Access class packages all corresponding security operating of key such as opening the key handle, sign verifying, encrypting, decrypting and storing the certification etc.

5) The class of KSP_Mutex and KSP_AuthUI

KSP_Mutex class is used to apply for and release the resources to prevent from deadlock or other unexpected results. KSP_AuthUI class is used to execute the related password authentication operations of USB.

In addition, KSP and CSP key container have a slightly difference in concept. In the CSP, a container could contain a lot of keys such as the encryption key pair, signature key pairs and symmetric keys etc., which usually depends on the designer who usually designs two pairs of keys that consist of the encryption key and signature key in a container in actual situation. But in KSP, each of the key containers could only have one key pair and does not make a difference of functions at all, and the key could be used in both verification and decryption.

3.2 The development of KSP based on USB key

Similarly, KSP can be implemented by either software or hardware. Relatively speaking, because KSP based on software implementation is related with storage and execution mechanism, it hardly has high security. KSP based on hardware implementation can bring us higher security, but also brings extra cost. So we can budget security and implementation cost by executing the most kernel security computing in the smart card, storing the user key and certificate and user certificate in it, and implementing the other function in the software outside smart card.

For the concern about program integrating, secondary development and convenience to use, we compile the code related with USB key's command into third party dynamic link library. The structure of Key container's storage in USB is showed in Figure.3:

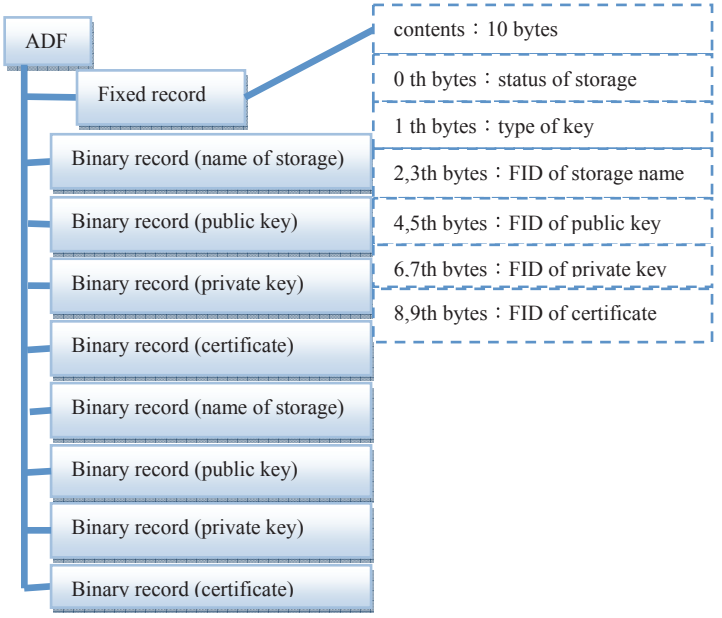

Fig. 3 The structure of Key container's storage

The 0th byte in the fixed-length record files is container's status which can represent that key container is empty, key has been established, container has been established, there is public key in container, there is a private key in container and a certificate accordingly. The first byte is key type, which can represent that whether this key is for key exchange or signature and the algorithm employed is RSA, ECDH or ECSDA.

3.3 Register of KSP

We need to execute the BCryptRegisterProvider() firstly to register KSPprovider, and then execute BCryptAddContextFunctionProvider() to increase the mechanism implemented in ksp.

The necessary steps to register ksp into a system are listed below: firstly copying the dll file that program generated to C:\WINDOWS\system32, and then employing BCryptRegisterProvider() to register Provider. This function uses the name of the provider and a point to a structure CRYPT_PROVIDER_REG which includes the register information of provider such as one or more aliases of this provider. And for the user mode provider, it also includes a point to CRYPT_IMAGE_REG. Alias can load a specific provider to employ a specific algorithm by application program CNG. For a cryptography algorithm provider, its alias can be passed by the parameter pszImplementation of BCryptOpenAlgorithmProvider(). For a KSP, its alias can be passed by the parameter pszProviderName of NCryptOpenStorageProvider(). If there are no aliases passed to these functions, default provider of specific algorithm will be loaded. The particular note is each register

provider need to have at least one alias, and this alias cannot be the same with the other provider's, or the register will fail.

For kernel mode provider, structure CRYPT_PROVIDER_REG also has a point to structure CRYPT_IMAGE_REG, but this member cannot be used, because the third party provider does not support to run under the kernel mode. User mode provider and kernel mode provider can only call BCryptRegisterProvider to register. The structure CRYPT_IMAGE_REG includes the path of provider and supporting crypto interface, such as encryption, hash and signature, etc.

After registration, one needs to call BCryptAddContextFunctionProvider() to specify algorithm it supports, and to call BCryptAddContextFunctionProvider() to execute registration for algorithms that each Provider supports.

We barely have information about KSP's certificate register, so we can register it in the same way as browser register of CSP certificate. During the registering, when CertSetCertificateContextProperty() needs to be executed, hCryptProv will be set in structure CERT_KEY_CONTEXT as handle of KSP provider. More details are:1) open a KSP Provider which is already registered by NCryptOpenStorageProvider(), 2) open a key pair which is already established in smart card, 3) append a context by CertCreateCertificateContext(), 4) set properties of a certificate by CertSetCertificateContextProperty(), 5) establish the link between private key and certificate by CertSetCertificateContextProperty(), 6) open system certificate storage area by CertAddCertificateContextToStore(), 7) load the context of certificate into system certificate storage area and 8) release all the opened resources.

3.4 AN application example

Here is an example of simple application based on the KSP as follows: using a email client system such as foxmail and an ECC certificate associated with KSP based on smart card to send a signed message in the vista system or above. Firstly ksp dll needs to be registered into the system; and then apply for a ECC certificate in the online authority certificate (A certificate is generated by a self-writing tool in our practical application) and associates with ksp dll, the e-mail address needs to be paid attention to in the application process that it should be the same with the sender's email address. After the configuration, open foxmail client and choose to send a signed email, the client would pop-up a "Select Certificate" dialog box, then select the certificate associated with the KSP dll, write a message and click "send", when receiving this message by foxmail client, the system registered KSP can automatically verify the data as legal or illegal signatures.

4. Conclusion

Based on introducing the architecture of KSP on the Next Generation Cryptography API, this paper discusses the important role of KSP, and a design and implementation of KSP is given based on smart card. Cryptographic functions can be integrated into IE, Foxmail, Outlook, Word and other software by KSP technology, and could also be applied to E-mail, E-commerce, E-government, Internet banking and other fields to meet their application requirements. KSP would be widely used and take the place of CSP for its good seamless connectivity and high-security features.

References

[1]Microsoft CryptoAPI and Cryptographic Service Providers.http://msdn.microsoft.com.

[2]Cryptography API: Next Generation. http://msdn.microsoft.com.

[3]S. Blake-Wilson, D. Brown, P. Lambert, Certicom Corp., Cosine Communications, “Use of Elliptic Curve Cryptography (ECC) Algorithms in Cryptographic Message Syntax(CMS),” RFC 3278, , April 2002.

[4]ANSI X9.62-1999,"The Elliptic Curve Digital Signature Algorithm (ECDSA), " American National Standard for Financial Services, Public Key Cryptography for the Financial Services Industry, January 7 1999.

[5]Koc CK,"High-speed RSA implementation,"RSA Laboratories Technical Report, TR 801, V. 1.0, 1996.

[6]Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, "RFC 3280, April 2002.

[7]Schaad, J., "Enhanced Security Services (ESS) Update: Adding CertID Algorithm Agility, "RFC 5035, August 2007.

[8]International Standard ISO/IEC 7816. Identification cards, *Integrated circuit(s) card with contacts*, 1997.